

NOIP2024模拟测试（4）

（2024年07月27日 8:00-12:00）

a

【题目描述】

有一个 n 个点的图(不保证联通，无重边无自环)，你可以把每条边染成黑色或白色，染好后，设只保留黑色边的联通块个数为 x ，设只保留白色边的联通块个数为 y ，求 $x + y$ 的最小值。

【输入格式】

从文件 a.in 中输入数据。

第一行两个数 n 和 m ，表示点数和边数。

接下来 m 行，每行两个数 x, y ，表示一条连接 x, y 的边。

【输出格式】

输出到文件 a.out 中。

一行一个字符串表示 t_{\max} 。

【样例输入1】

```
1 | 3 3
2 | 1 2
3 | 2 3
4 | 3 1
```

【样例输出1】

```
1 | 3
```

【样例输入2】

见下发文件 ex_a2.in

【样例输出2】

见下发文件 ex_a2.ans

【数据范围】

对于30%的数据： $m, n \leq 15$

另有30%的数据：图是一条链

对于所有数据： $m, n \leq 10^5$

内存限制： $512MiB$

时间限制： $1000ms$

b

【题目描述】

给你一个字符串 s ，你可以重排 s 中的字母而得到字符串 t 。定义 t_{\max} 为 t 和 $\text{rev}(t)$ ($\text{rev}(t)$ 表示把 t 前后翻转得到的字符串) 中字典序较大者。现在你需要让 t_{\max} 的字典序最小，并输出 t_{\max} (而不是 t ，所以答案是唯一的)。

【输入格式】

从文件 b.in 中输入数据。

一行一个字符串表示 s 。

【输出格式】

输出到文件 b.out 中。

一行一个字符串表示 t_{\max} 。

【样例输入1】

```
1 | abb
```

【样例输出1】

```
1 | bab
```

【样例输入2】

```
1 | abbcc
```

【样例输出2】

```
1 | bbcca
```

【样例输入3】

```
1 | ffcaba
```

【样例输出3】

1 | acffba

【样例输入4】

1 | aaaabbcc

【样例输出4】

1 | aabccbaa

【数据范围】

对于20%的数据： $|s| \leq 10$

另有30%的数据： s 中每种字母个数均为偶数

另有30%的数据： s 中只含有 a 和 b

对于所有数据： $|s| \leq 10^6$

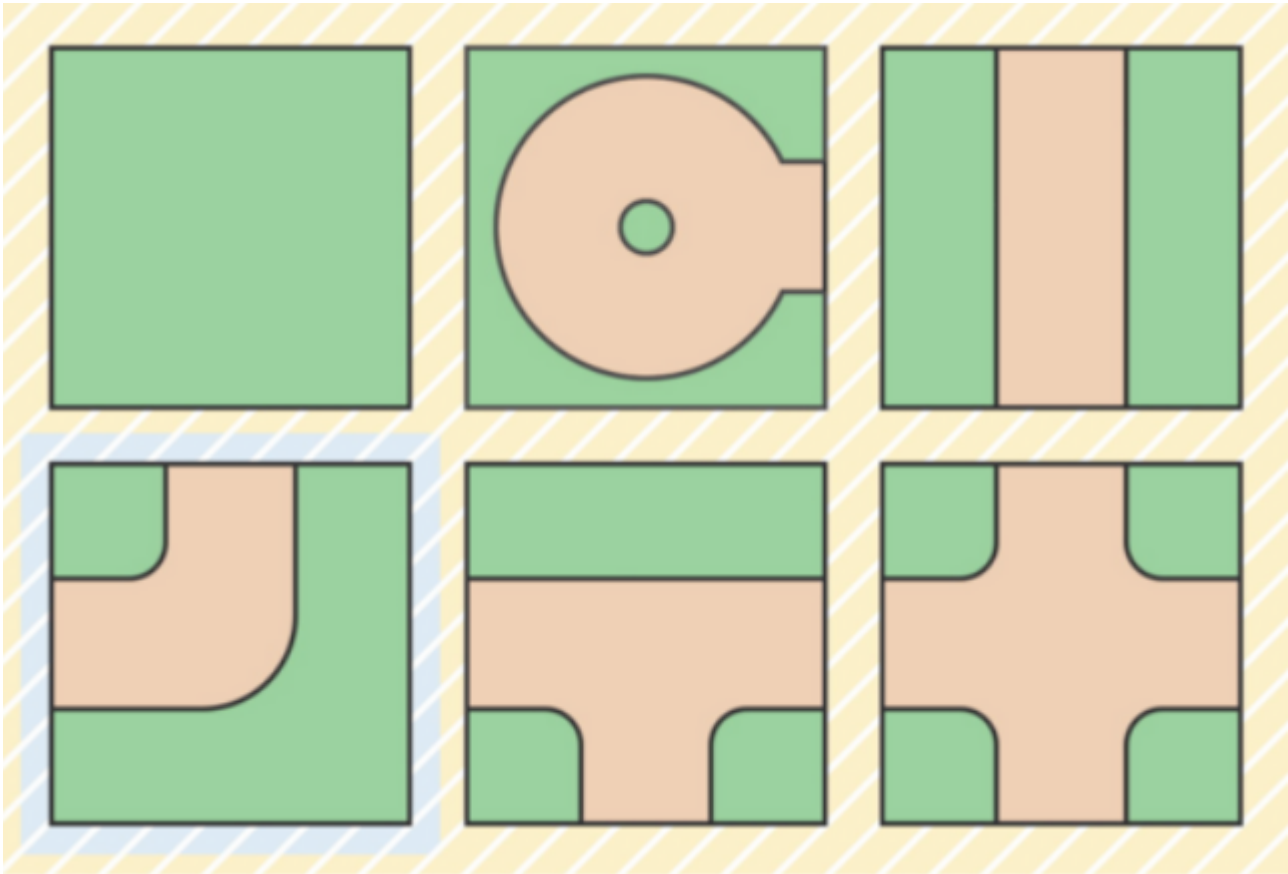
内存限制： $512MiB$

时间限制： $1000ms$

C

【题目描述】

你来到一家修地下水管的公司实习，你只会修一种水管，但公司里的其他员工都会修 6 种。



上图中边缘是蓝色的那种就是你唯一会修的水管（所有修法均可以旋转，左上角的全空的那种修法也算一种，只是你不会）。

打工的第一天，你和同事们要去修一个 H 行 W 列的网格中的水管，其中有一些水管已经被偷走了。

现在公司下了指令，对于所有水管被偷走的方格，规定你在一些方格上必须修水管，在一些方格上不能修水管，在一些方格上可修可不修。

你决定修尽量多的水管，但是你必须保证在你修完后，存在一种在其他你没有修水管并且水管被偷的方格上修那 6 种水管之一的方案，使得不漏水（即不存在一个水管的口对着的不是另一个水管的口，边界也不行）。

由于你修水管还要花时间，所以你只有 1s 的时间来决定修哪些水管。

【输入格式】

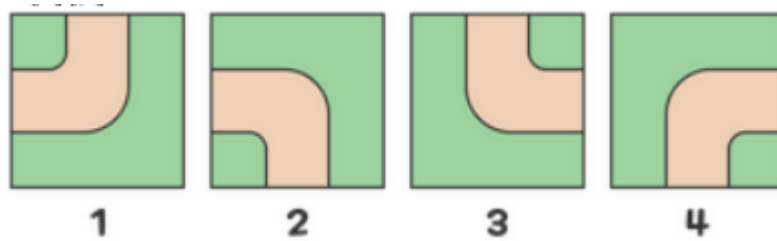
从文件 c.in 中读入数据

第一行两个数 H, W 。

接下来输入一个 H, W 的字符数组 a ，表示每个网格上的状态。

对于网格的第 i 行第 j 列,数组 a 的第 i 行第 j 列表示它的状态：

- $a[i][j] = 1, 2, 3, 4$: 第 i 行第 j 列存在一个形状为 $a[i][j]$ 的水管没有被偷。



- $a[i][j] = 'o'$: 公司规定你必须在这个格子修水管。
- $a[i][j] = 'x'$: 公司规定你不能在这个格子修水管。
- $a[i][j] = '.'$: 这个格子可以修水管也可以不修。

####

- 【输出格式】

输出到文件 c.out 中。

输出一行一个整数，表示没被偷走的水管数量加上你新修的水管数量的最大值。

特别的，如果不存在一种方案，输出 -1。

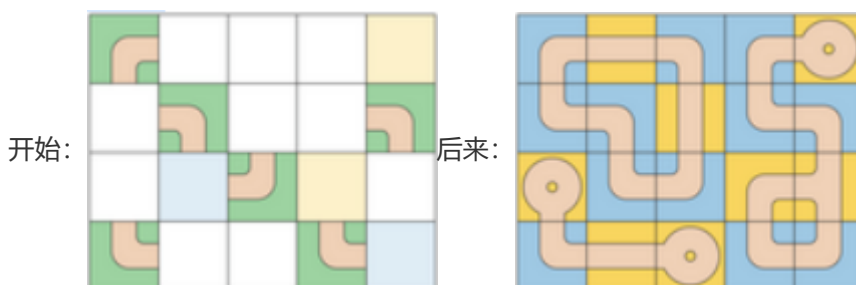
【样例输入1】

```
1 4 5
2 4...x
3 .2..2
4 .o1x.
5 3..3o
```

【样例输出1】

```
1 12
```

【样例1解释】



【样例输入2】

```
1 2 3
2 4o2
3 3x1
```

【样例输出2】

```
1 -1
```

【样例输入3】

```
1 3 3
2 ...
3 3o.
4 .ox
```

【样例输出3】

```
1 5
```

【样例输入4】

见下发文件 ex_c4.in

【样例输出4】

见下发文件ex_c4.ans

【数据范围】

对所有数据 $1 \leq H, W \leq 100$ 。

子任务编号	分 数	特殊性质
1	10	$H = W = 100$, $a[i][j]$ 在 $1, 2, 3, 4, ., o, x$ 中随机
2	30	$H, W \leq 3$
\$	30	$a[i][j] = '!$
4	20	$a[i][j] \in \{., o, x\}$
5	10\$	无

内存限制：512MiB

时间限制：1000ms

d

【题目描述】

显示器上有 5 个数位，每个位置可以显示 0 到 9，也就是说显示器可以显示 0 到 99999 的这些数字。初始所有位置都是 0，系统从 $[L, R]$ 中随机选出一个数字 x ，现在我们要猜这个数字是几。

每一步，我们可以干两件事情中的一件：

- 1. 修改一个数位；
- 2. 问问当前数字和 x 的大小关系（返回大于、等于、小于）；

当我们确定 x 是几的时候，我们就 win 了。

现在我们要设计一个最优策略，请问我们最少需要多少步，才能猜到所有的 x ？

【输入格式】

从文件 d.in 中读入数据。

第一行一个数表示数据组数 T 。

对于每组数据，一行两个整数 L, R 。

【输出格式】

输出到文件 d.out 中。

对于每组数据，一行一个整数表示答案。

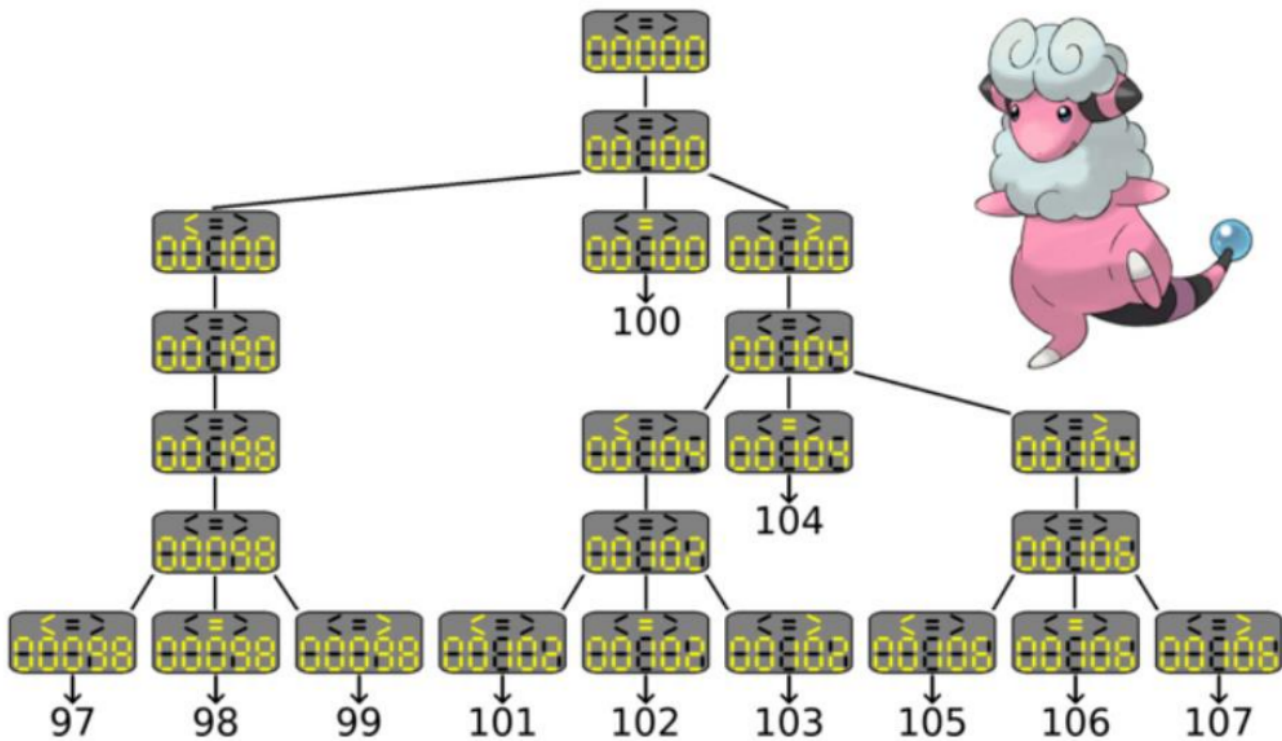
【样例输入1】

1	3
2	97 107
3	12043 12045
4	61 69

【样例输出1】

1	6
2	5
3	7

【样例1解释】



【样例输入2】

见下发文件 ex_d2.in

【样例输出2】

见下发文件ex_d2.ans

【数据范围】

$1 \leq T \leq 50, 1 \leq L < R \leq 99999$

测试点	R<	其他条件
1~4	10	无
5~8	100	无
9~10	100000	$R - L < 10$
11~12	100000	$R - L < 100$
13~20	100000	无

内存限制：512MiB

时间限制：8000ms